Chikezie Ogbogu

Brooklyn Igbojionu

Kian Sharma

(Group 49)

Cs335 Group Project on Software Engineering

Assigned Theme: Transport

Our application:

LiveViewer will be an application that enables users to plan transportation ahead of time, each trip will be personalized to each user and ensure they can arrive at there destination in a timely fashion.

Features:

- **Live tracking - LiveViewer** will provide real time tracking of transportation services such as buses or trains to provide a better and real time updated estimate of a user's journey time.
- **Late Tracking -** LiveViewer will collect data on how long users wait for various transport services and factor this into its transport recommendation, advising users of when a service will arrive late or arrive early.
- **Social Comparison-** LiveViewer will have a social setting where users who have taken similar trips to the same destination (eg; Maynooth University to Bachelors Walk) will have there trip uploaded and users can compare which route is the fastest to take at that particular hour.
- **Push Notifications-** LiveViewer will notify users when its time to leave to get to there destination in time, it will learn from user habits to provide even more relevant notifications
- L**ast Mile Ride-Sharing-** LiveViewer will have an option for users who are travelling to the same place at the same time where public transport might not reach to instead share a taxi or a private car reducing the cost for each user.

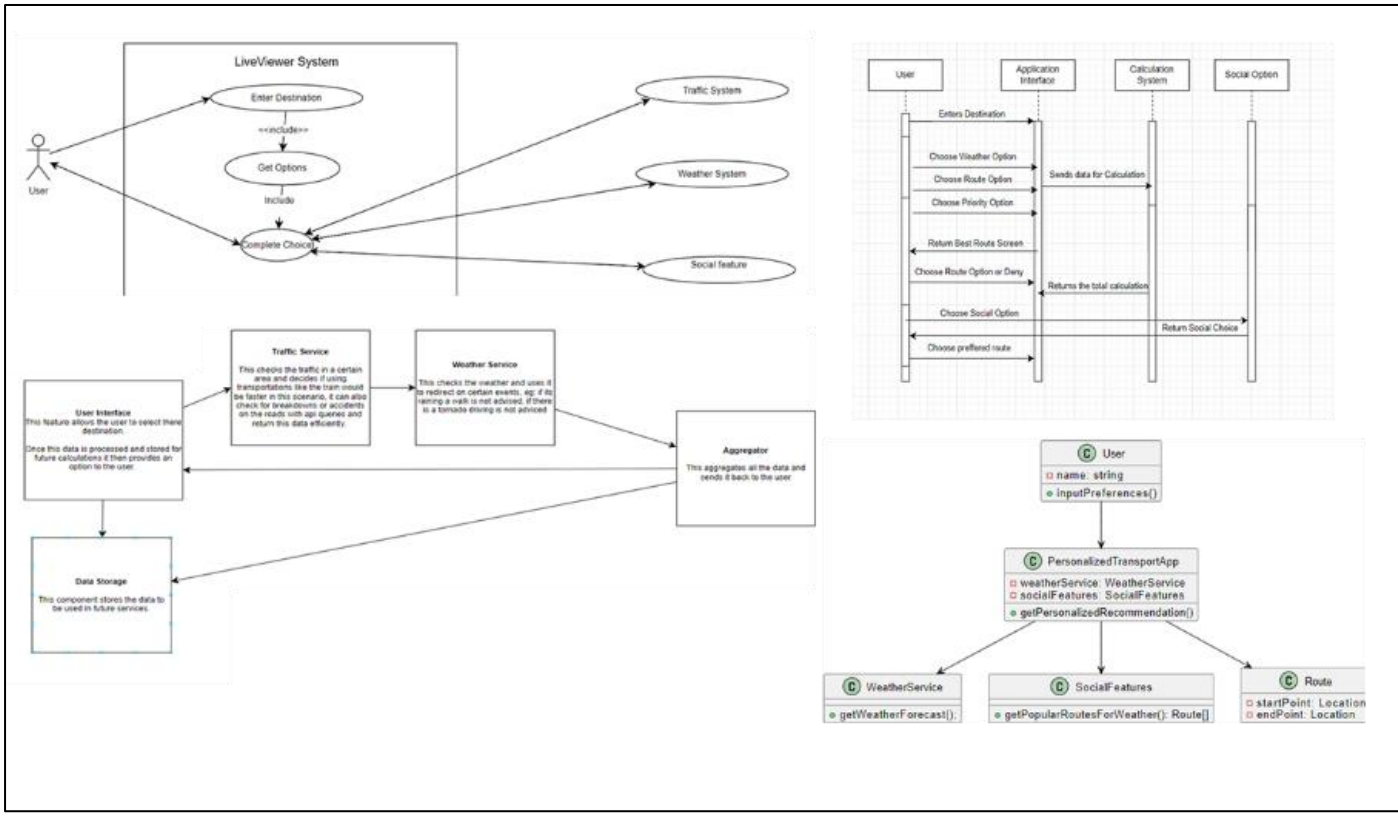
The customer has come to you with a vague specification. In this project you are required to: 1. You must elicit a substantial set of software requirements from them in the form of User Stories (10 User Stories should be created).

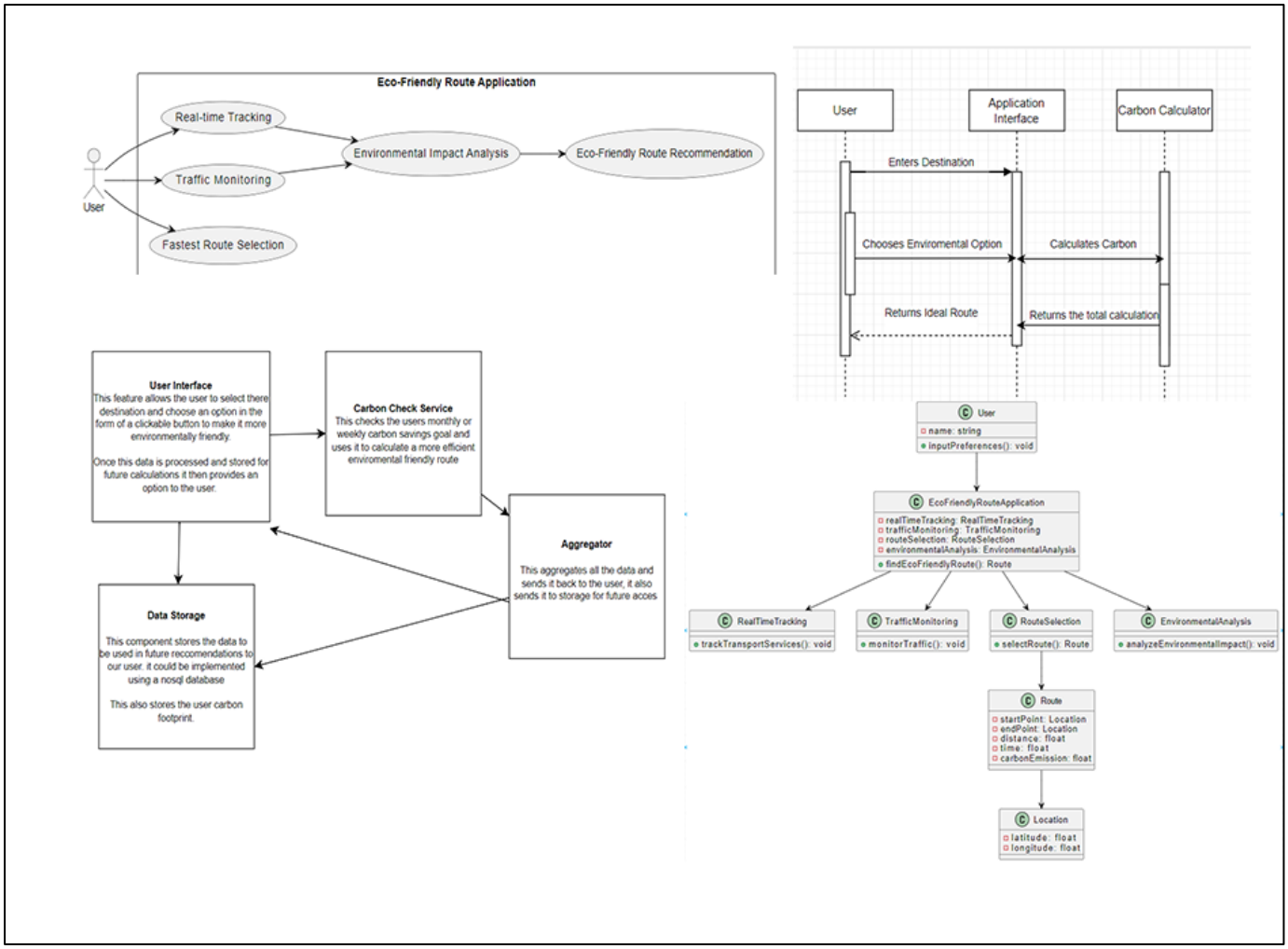
1. As a user I want to be able to get to my destination as quickly as possible, I would like to be able to take the shortest route every time to get to my destination, I imagine an application that would assist me in this by providing real time tracking of transport services and would automatically know when I choose the fastest option possible to redirect me away from services that are known to come late or not come at all. This application should also check with the traffic management systems in my location to make sure I do not take routes that are in heavy traffic reducing my speed to get to my destination. In my search of the fastest way to an event I would imagine an application that would disregard pricing and simply show me the quickest solution, whether it is a nearby bus or a taxi, the application would prioritise speed over economical savings.
2. As a user I care deeply about the environment and I expect my application to have a feature to cater to this, I have come to understand that transport is a significant contributor to carbon emissions and I would like to reduce my carbon footprint while moving from one location to the next, I would expect a feature that caters to this need, comparing all the different routes to get to my destination and providing the most envinromental friendly route that does not endanger the environment.

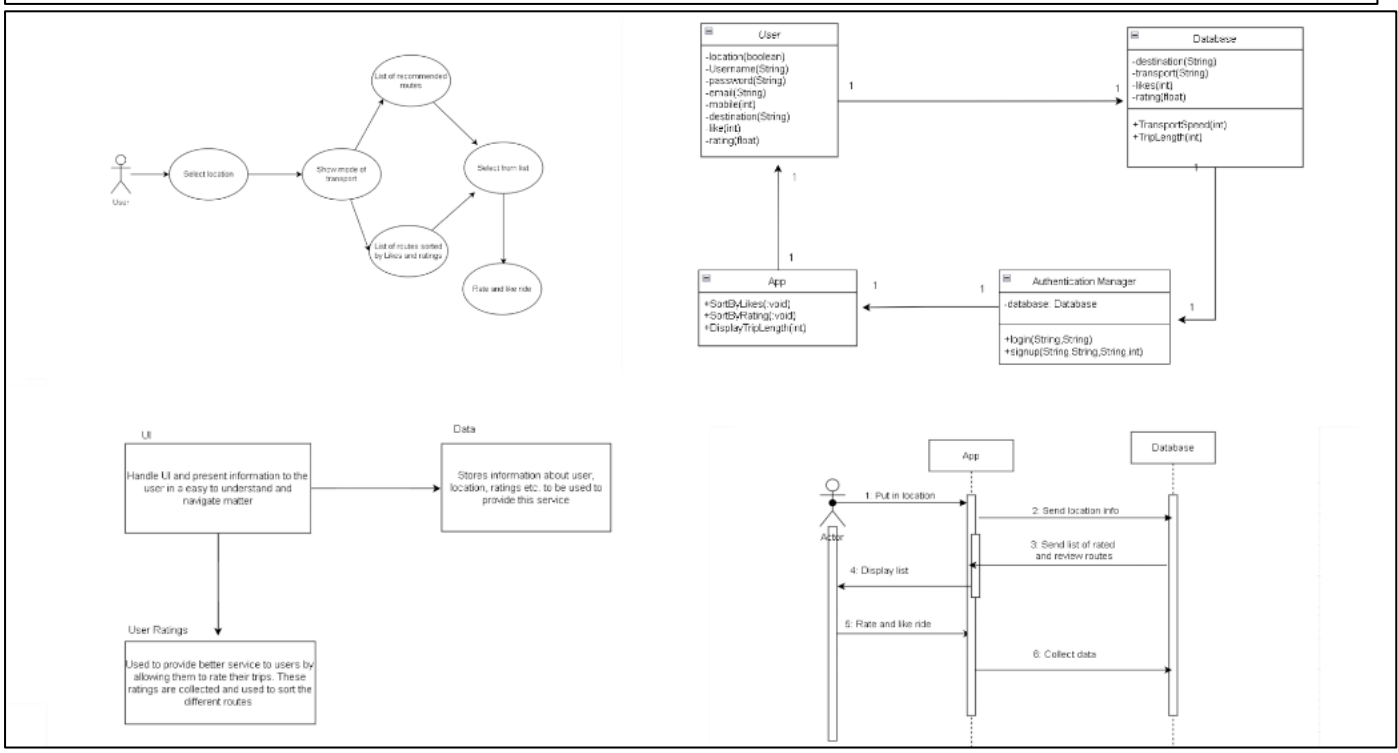
3. As a user I expect my application to offer me personalized transport solutions that is catered directly to my situation, my application should have a feature that understands that although it was bright and sunny yesterday and a route was walkable, that is not the case today and it would reduce my speed in getting to my destination if I had to walk in the snow or the rain, this feature will check the weather at my location any time I interact with my application and provide suitable recommendations based on this weather. It will also interact with its social features to understand what different users have selected in this particular weather.
4. As a user I expect my application to understand that the journey to some routes might include multiple modes of transports, I would expect a feature that shows that during rush hour times trains might be quicker and during normal times buses might be faster. This feature will provide me with different variations of getting to my destination in a timely manner even if it includes taking a bus, walking and catching a train to avoid traffic.
5. As a user I expect a feature that understands that often getting to rural routes is hard without a primary car. My application should be able to connect me to multiple users who also are looking to travel to the same destination as me at a similar time and offer a feature for us all to contribute towards ride sharing so it is cheaper and more efficient for all of us.
6. As a user when I want to travel to a destination I want to be able to look at the journeys of other people who have travelled to the same destination to understand which option is best, although the application will provide me with a recommended route to begin through its pathfinding algorithm, I believe it is necessary for me to be able to view how other users who could be locals have travelled to a destination as locals will know an area better than a computer algorithm will and there journeys might assist me in knowing which services or which routes that although the appear good on paper might not be practical in a real world scenario. This social aspect should let users vote through a liking system to provide “most liked” and “top rated” user journey routes that can then be compared to the algorithms recommendation by me. This will be especially useful in scenic routes or exploring the great outdoors.
7. As a user I want to be able to plan shared trips with my friends, I want all of us to be able to pick and choose which route we should take, for example if me and my friends In our university want to go out to eat at a certain time I expect the application to be able to plan our journey at that time, the application should then make sure all of us agree to this plan and then provide live updates as different friends might opt different routes that cater to there schedules better it should enable us to track each others location and provide an estimated time of when everyone will be present at our destination as part of its social function.
8. As a user I expect my application to let me plan trips far ahead of time, for example if I have a journey to the airport in a month from now, I should be able to plan how I will get to the airport a month in advance and when the dates come closer my application should provide me with notifications alerting me of how I will travel and my estimated time of arrival.
9. As a user if I opt to take the public transport method and in some cases this might lead me to taking multiple services I need a feature that simplifies ticket purchasing for me, there should be a in house feature that helps me buy the tickets for all these services at the same time with the user information I already purchased. It should provide an aggregate price and enable me to pay all at once then the system can purchase each individual tickets and when it comes time to use the tickets in a transport switching scenario the tickets should be available automatically and I shouldn’t have to fumble and search for the tickets.

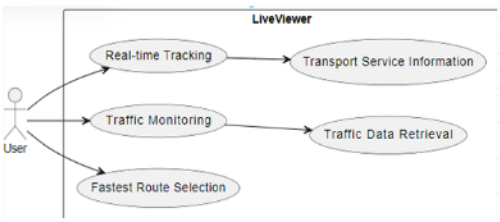
10. As a user I fully expect my system to be fully catered to me, my transport companion should live beside me and learn my habits and adapt to them ensuring I always reach my destination in a timely manner, if I am likely to miss my transport services as I have the previous times my transport companion should learn from this and give me alerts to begin earlier than normal. If I consistently arrive late my application should refer me to a earlier service instead so it doesn't happen more often and so on and so forth.

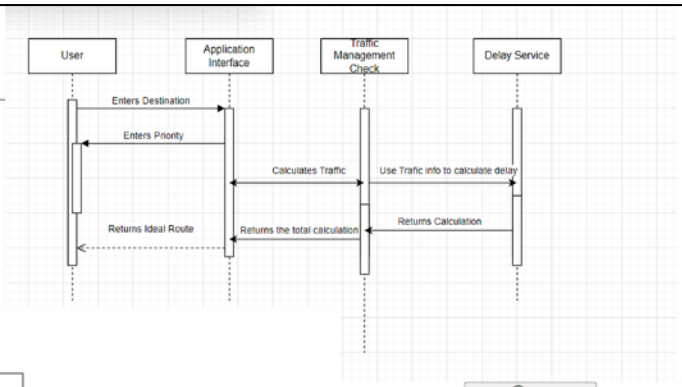
2. The User Stories should then be translated into formal descriptions/models that can be used by all project stakeholders, most importantly the developers and testers. These models would include a complete set of UML diagrams: Use Case, Sequence Diagram, Class Diagram and Architecture Diagram

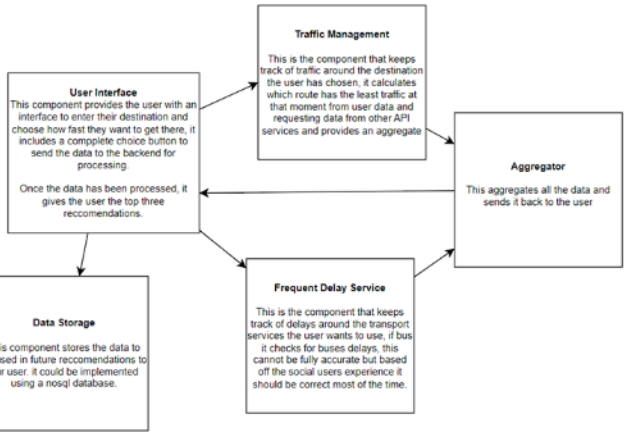


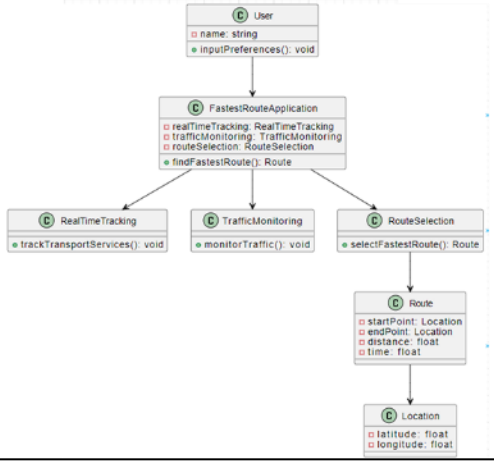

Eco-Friendly Route Application
Real-time Tracking
Traffic Monitoring
Fastest Route Selection
Environmental Impact Analysis
Eco-Friendly Route Recommendation
User
User
Application Interface
Carbon Calculator
Enters Destination
Chooses Enviromental Option
Calculates Carbon
Returns Ideal Route
Returns the total calculation
User Interface
This feature allows the user to select there destination and choose an option in the form of a clickable button to make it more environmentally friendly.
Once this data is processed and stored for future calculations it then provides an option to the user.
Carbon Check Service
This checks the users monthly or weekly carbon savings goal and uses it to calculate a more efficient enviromental friendly route
Aggregator
This aggregates all the data and sends it back to the user, it also sends it to storage for future acces
Data Storage
This component stores the data to be used in future reccomendations to our user. it could be implemented using a nosql database
This also stores the user carbon footprint.
User
name: string
inputPreferences(): void
EcoFriendlyRouteApplication
realTimeTracking: RealTimeTracking
trafficMonitoring: TrafficMonitoring
routeSelection: RouteSelection
environmentalAnalysis: EnvironmentalAnalysis
findEcoFriendlyRoute(): Route
RealTimeTracking
trackTransportServices(): void
TrafficMonitoring
monitorTraffic(): void
RouteSelection
selectRoute(): Route
EnvironmentalAnalysis
analyzeEnvironmentalImpact(): void
Route
startPoint: Location
endPoint: Location
distance: float
time: float
carbonEmission: float
Location
latitude: float
longitude: float

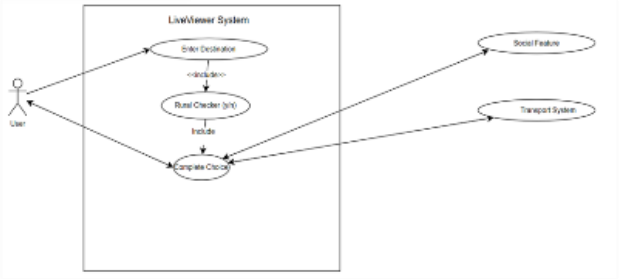

User
Select location
Show mode of transport
List of recommended routes
Select from list
List of routes sorted by Likes and ratings
Rate and like ride
User
-location(boolean)
-Username(String)
-password(String)
-email(String)
-mobile(int)
-destination(String)
-like(int)
-rating(float)
DataBase
-destination(String)
-transport(String)
-likes(int)
-rating(float)
+TransportSpeed(int)
+TripLength(int)
App
+SortByLikes(:void);
+SortByRating(:void);
+DisplayTripLength(int)
Authentication Manager
-database: Database
+login(String,String)
+signup(String,String,int)
UI
Handle UI and present information to the user in a easy to understand and navigate matter
Data
Stores information about user, location, ratings etc. to be used to provide this service
User Ratings
Used to provide better service to users by allowing them to rate their trips. These ratings are collected and used to sort the different routes
App
Database
Actor
1: Put in location
2: Send location info
3: Send list of rated and review routes
4: Display list
5: Rate and like ride
6: Collect data

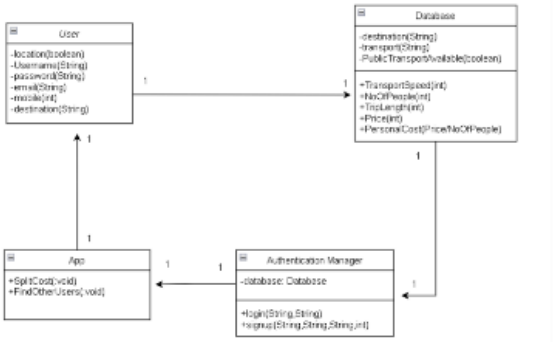

LiveViewer
Real-time Tracking
Transport Service Information
Traffic Monitoring
Traffic Data Retrieval
Fastest Route Selection
User

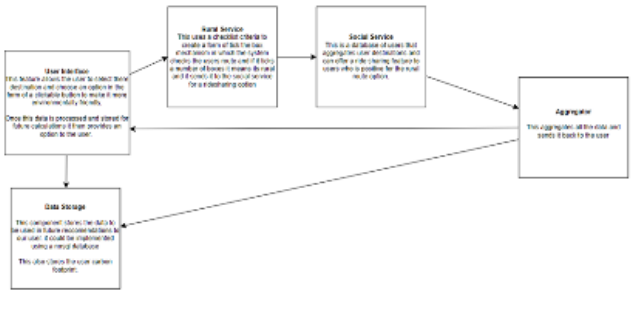

User
Application Interface
Traffic Management Check
Delay Service
Enters Destination
Enters Priority
Calculates Traffic
Use Trafic info to calculate delay
Returns Ideal Route
Returns the total calculation
Returns Calculation

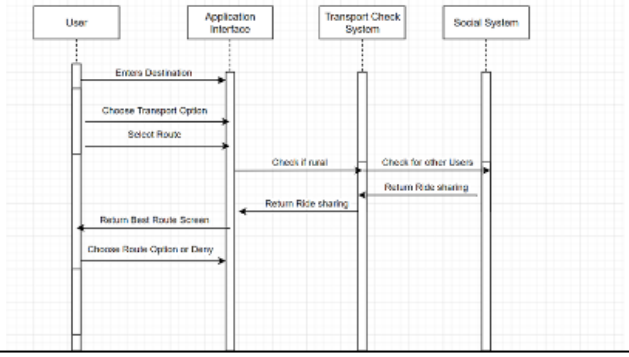

User Interface
This component provides the user with an interface to enter their destination and choose how fast they want to get there, it includes a compplete choice button to send the data to the backend for processing.
Once the data has been processed, it gives the user the top three reccomendations.
Traffic Management
This is the component that keeps track of traffic around the destination the user has chosen, it calculates which route has the least traffic at that moment from user data and requesting data from other API services and provides an aggregate
Aggregator
This aggregates all the data and sends it back to the user
Data Storage
This component stores the data to be used in future reccomendations to our user. it could be implemented using a nosql database.
Frequent Delay Service
This is the component that keeps track of delays around the transport services the user wants to use, if bus it checks for buses delays, this cannot be fully accurate but based off the social users experience it should be correct most of the time.

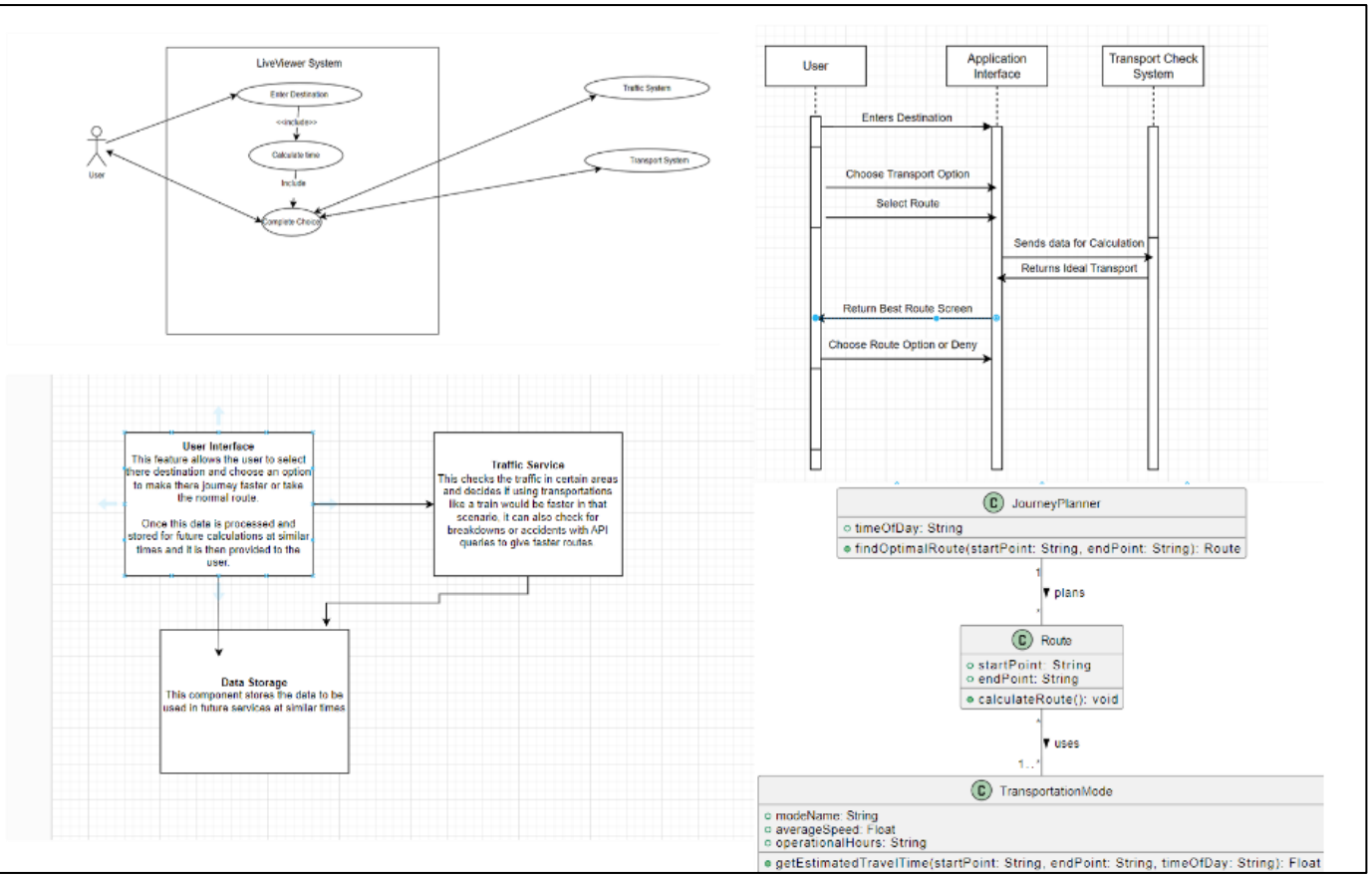

User
name: string
inputPreferences(): void
FastestRouteApplication
realTimeTracking: RealTimeTracking
trafficMonitoring: TrafficMonitoring
routeSelection: RouteSelection
findFastestRoute(): Route
RealTimeTracking
trackTransportServices(): void
TrafficMonitoring
monitorTraffic(): void
RouteSelection
selectFastestRoute(): Route
Route
startPoint: Location
endPoint: Location
distance: float
time: float
Location
latitude: float
longitude: float

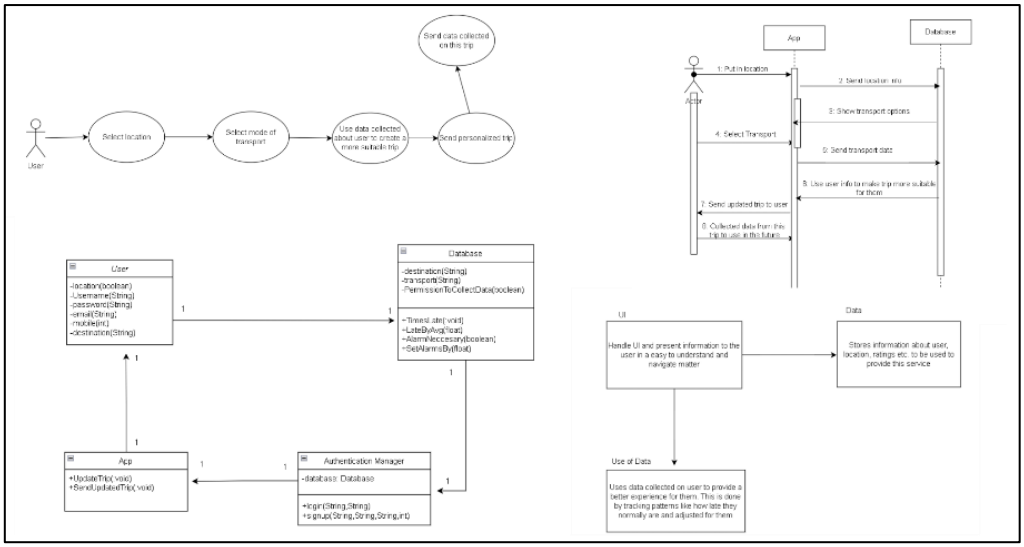

LiveViewer System
Enter Destination
<<include>>
Rural Checker (s/n)
Include
Complete Choice
Social Feature
Transport System
User

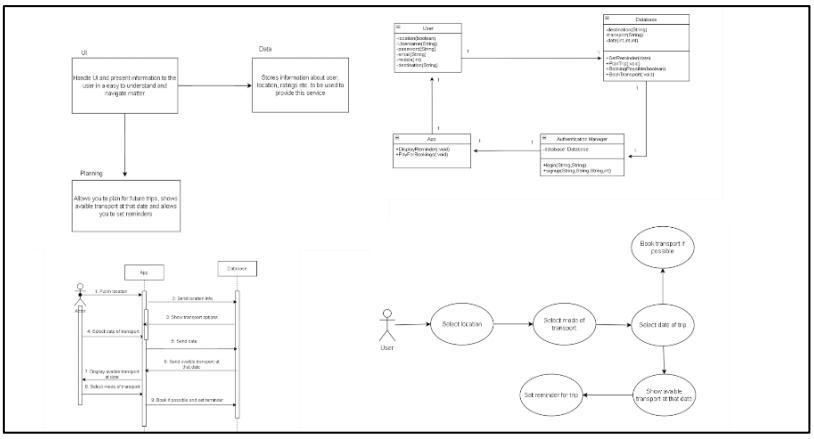

User
-location(boolean)
-Username(String)
-password(String)
-email(String)
-mobile(int)
-destination(String)
Database
-destination(String)
-transport(String)
-PublicTransportAvailable(boolean)
+TransportSpeed(int)
+NoOfPeople(int)
+TripLength(int)
+Price(int)
+PersonalCost(Price/NoOfPeople)
App
+SplitCost(:void)
+FindOtherUsers(:void)
Authentication Manager
-database: Database
+login(String,String)
+signup(String,String,String,int)

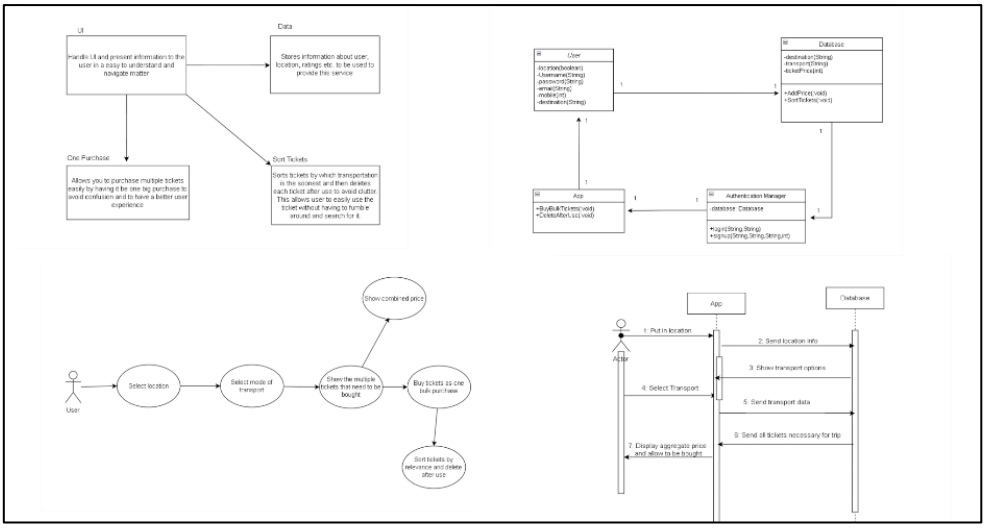

Rural Service
Social Service
Aggregator
Data Storage

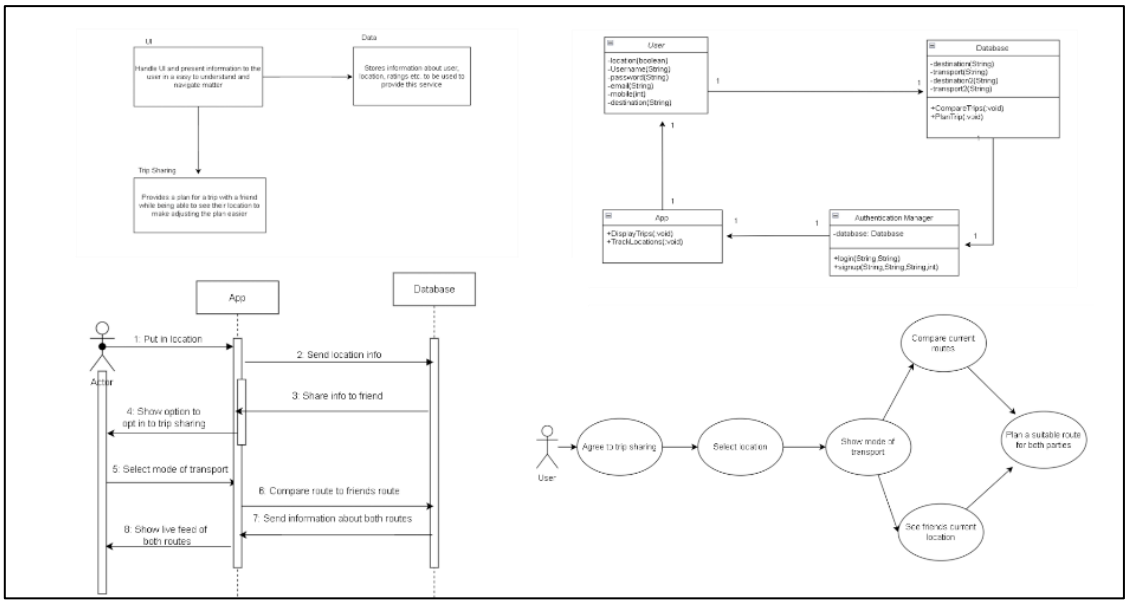

User
Application Interface
Transport Check System
Social System
Enters Destination
Choose Transport Option
Select Route
Check if rural
Check for other Users
Return Ride sharing
Return Ride sharing
Return Best Route Screen
Choose Route Option or Deny


LiveViewer System
Enter Destination
<<include>>
Calculate time
Include
Complete Check
User
Traffic System
Transport System
User
Application Interface
Transport Check System
Enters Destination
Choose Transport Option
Select Route
Sends data for Calculation
Returns Ideal Transport
Return Best Route Screen
Choose Route Option or Deny
User Interface
This feature allows the user to select there destination and choose an option to make there journey faster or take the normal route.
Once this data is processed and stored for future calculations at similar times and it is then provided to the user.
Traffic Service
This checks the traffic in certain areas and decides if using transportations like a train would be faster in that scenario. it can also check for breakdowns or accidents with API queries to give faster routes.
Data Storage
This component stores the data to be used in futura services at similar times
JourneyPlanner
timeOfDay: String
findOptimalRoute(startPoint: String, endPoint: String): Route
1
plans
*
Route
startPoint: String
endPoint: String
calculateRoute(): void
*
uses
1..*
TransportationMode
modeName: String
averageSpeed: Float
operationalHours: String
getEstimatedTravelTime(startPoint: String, endPoint: String, timeOfDay: String): Float


Send data collected on this trip
Select location
Select mode of transport
Use data collected about user to create a more suitable trip
Send personalized trip
User
App
Database
Actor
1: Put in location
2: Send location info
3: Show transport options
4: Select Transport
5: Send transport data
6: Use user info to make trip more suitable for them
7: Send updated trip to user
8: Collected data from this trip to use in the future
User
-location(boolean)
-Username(String)
-password(String)
-email(String;
-mobile(int)
-destination(String)
Database
-destination(String)
-transport(String)
-PermissionToCollectData(boolean)
+TimesLate(void)
+LateByAvg(float)
+AlarmNeccessary(boolean)
-SetAlarmsBy(float)
App
+UpdateTrip(void)
+SendUpdatedTrip(void)
Authentication Manager
-database: Database
+login(String,String)
+signup(String,String,String,int)
UI
Handle UI and present information to the user in a way to understand and navigate matter
Data
Stores information about user, location, ratings etc. to be used to provide this service
Use of Data
Uses data collected on user to provide a better experience for them. This is done by tracking patterns like how late they normally are and adjusted for them


UI
Handle UI and present information to the user in a way to understand and navigate matter
Data
Stores information about user, location, ratings etc. to be used to provide this service
Planning
Allows you to plan for future trips, shows avaible transport at that date and allows you to set reminders
User
Database
App
Authentication Manager
Book transport if possible
Select location
Select mode of transport
Select date of trip
Set reminder for trip
Show avaible transport at that date


UI
Handle UI and present information to the user in a easy to understand and navigate matter
Data
Stores information about user, location, ratings etc. to be used to provide this service
One Purchase
Allows you to purchase multiple tickets easily by having it be one big purchase to avoid confusion and to have a better user experience
Sort Tickets
Sorts tickets by which transportation is the soonest and then deletes each ticket after use to avoid clutter. This allows user to easily use the ticket without having to fumble around and search for it.
User
-location(boolean)
-Username(String)
-password(String)
-email(String)
-mobile(int)
-destination(String)
Database
-destination(String)
-transport(String)
-ticketPrice(int)
+AddPrice(:void)
+SortTickets(:void)
App
+BuyBulkTickets(:void)
+DeleteAfterUse(:void)
Authentication Manager
-database: Database
+login(String,String)
+signup(String,String,String,int)
Select location
Select mode of transport
Show the multiple tickets that need to be bought
Show combined price
Buy tickets as one bulk purchase
Sort tickets by relevance and delete after use
Actor
1: Put in location
2: Send location info
3: Show transport options
4: Select Transport
5: Send transport data
6: Send all tickets necessary for trip
7: Display aggregate price and allow to be bought


UI
Handle UI and present information to the user in a easy to understand and navigate matter
Data
Stores information about user, location, ratings etc. to be used to provide this service
Trip Sharing
Provides a plan for a trip with a friend, while being able to see the r location to make adjusting the plan easier
User
-location(boolean)
-Username(String)
-password(String)
-email(String)
-mobile(int)
-destination(String)
Database
-destination1(String)
-transport1(String)
-destination2(String)
-transport2(String)
+CompareTrips(:void)
+PlanTrip(:void)
App
+DisplayTrips(:void)
+TrackLocations(:void)
Authentication Manager
-database: Database
+login(String,String)
+signup(String,String,String,int)
Actor
1: Put in location
2: Send location info
3: Share info to friend
4: Show option to opt in to trip sharing
5: Select mode of transport
6: Compare route to friends route
7: Send information about both routes
8: Show live feed of both routes
User
Agree to trip sharing
Select location
Show mode of transport
Compare current routes
Plan a suitable route for both parties
See friends current location

3. A mock-up of the user interface should be produced.

4. Additionally, an outline set of System tests that can be used to validate the User requirements should be created.

Test Cases

1. Shortest Route

- Test the accuracy of the system in calculating the shortest route based on a user's inputted location.
- Test the system's ability to respond in real time to events such as crashes, traffic, and delays, and notify and redirect the user based on this information.
- Test the system to ensure it is recommending the fastest route available to the user first, irrespective of pricing.

2. Carbon Footprint

- Test the systems to ensure that it is recommending environmentally friendly means of transport to the users inputted destination.
- Test the accuracy and validity of the information the system is acting upon.

3. Weather

- Test the system's ability to recommend routes based on real time weather information and recommend more suitable means of travel depending on the weather, based on the users destination.
- Test the accuracy and validity of the systems recommendations and weather information.

4. Busy Times

- Test the system's ability to adapt to changing circumstances based on time, and notify and redirect users from busy or late means of transport based on the users inputted route
- Test the system to ensure the system is accurately keeping track of wait times, delays, and busy times of day.

5. Ride Share

- Test the system's ability to find and group users who have inputted a rural, hard to reach destination and recommend them a shared means of travel to their respective locations based on the users' input.
- Test the system to ensure it is recommending an efficient route for all grouped users and providing them with the means to contribute collectively to lower costs for each respective individual.
- Test the system to ensure the accuracy and validity of the users and routes it is grouping together, and ensure the efficiency of the shared route for each user.

Contributions-

Chikezie Ogbogu

- **Diagrams for User Stories 6-10, Power point**

Brooklyn Igbojionu

- **User Stories, Diagrams for User Stories 1-5, Video**

Kian Sharma

- **Wireframes and System Tests**